## Appendix A Experiment results

This section serves as a complementary analysis to the experimental results of the training process presented in the main text. We systematically compare the pre-training performance of all baseline methods and their post-fine-tuning performance across multiple randomized trials. Results are organized in Tables A1, A2 and A3, which includes: Pre-training metrics and fine-tuned performance under varying observation loss probabilities, evaluated through three independent random seeds to ensure statistical robustness.

**Table A1** Pre-trained joint training results. During training, each map and task has a fixed Maximum Episode Steps (MES). After training for a set number of real environment interaction steps (REIS) across different maps or tasks, the model weights are saved. Performance is then evaluated over 1000 test episodes, measuring the average win rate (SMAC) or step return (MAMuJoCo) and their standard deviations, with the highest average performance highlighted in bold.

| Maps/Task | REIS | MES | RMIOv2 | RMIO [24] | MAG [19] | MAMBA [18] | MAPPO [49] | QMIX [7] | FACMAC [27] |
|---|---|---|---|---|---|---|---|---|---|
| *2s_vs_1sc* | 15k | 300 | **93.4(3.1)** | 90.4(5.4) | 86.8(4.4) | 63.8(15.2) | 0.0(0.0) | 0.0(0.0) | 0.0(0.0) |
| *3s_vs_3z* | 50k | 150 | **95.2(2.7)** | 88.2(3.6) | 83.2(5.6) | 78.1(10.2) | 0.0(0.0) | 0.0(0.0) | 0.0(0.0) |
| *2s3z* | 80k | 120 | **98.1(1.1)** | 87.1(3.2) | 67.1(11.2) | 70.9(11.9) | 0.0(0.0) | 23.1(3.7) | 7.6(1.2) |
| *3s_vs_4z* | 200k | 200 | **96.9(0.8)** | 96.1(1.2) | 81.3(11.8) | 64.4(32.0) | 0.0(0.0) | 0.0(0.0) | 0.0(0.0) |
| *3s_vs_5z* | 300k | 250 | **93.4(1.8)** | 85.3(3.4) | 55.1(12.4) | 53.3(8.5) | 0.0(0.0) | 0.0(0.0) | 0.0(0.0) |
| *1c3s5z* | 75k | 180 | **97.6(2.8)** | 84.9(3.6) | 64.9(12.9) | 53.7(9.2) | 0.0(0.0) | 21.0(4.3) | 6.9(0.7) |
| *8m* | 40k | 120 | **90.3(2.2)** | 83.4(3.8) | 63.3(7.8) | 36.6(7.4) | 0.0(0.0) | 28.3(6.0) | 17.4(2.1) |
| *corridor* | 400k | 400 | **85.7(3.6)** | 60.2(17.6) | 27.4(7.3) | 39.3(9.2) | 0.0(0.0) | 0.0(0.0) | 0.0(0.0) |
| 6 agents *HalfCheetah* | 200k | 300 | **5.08(0.46)** | 4.63(0.43) | 4.07(0.37) | 3.71(0.69) | 0.45(0.38) | / | 1.62(0.24) |
| 4 agents *Swimmer* | 200k | 300 | 0.32(0.01) | **0.34(0.01)** | 0.29(0.01) | 0.26(0.01) | 0.09(0.02) | / | 0.17(0.03) |
| 10 agents *Swimmer* | 200k | 300 | **0.35(0.02)** | 0.34(0.02) | 0.14(0.19) | 0.22(0.07) | -0.15(0.02) | / | -0.12(0.02) |
| 4 agents *Ant* | 200k | 300 | **2.44(0.42)** | 2.13(0.35) | 1.88(0.55) | 1.31(0.23) | 0.46(0.03) | / | 0.67(0.02) |

**Table A2** The experimental results of all methods in 1000 random matches under different observation loss probabilities $p_{\text{loss}}$ on different maps in SMAC. Each map name is followed by the number of fine-tuning steps in parentheses (e.g., $8m$ (15k)). Results are reported as the average win rate percentage (standard deviation) aggregated over three random seeds, with the highest-performing entry highlighted in bold.

| Map | $p_{\text{loss}}$ | RMIOv2 | RMIO [24] | MAG [19] | MAMBA [18] | Map | $p_{\text{loss}}$ | RMIOv2 | RMIO [24] | MAG [19] | MAMBA [18] |
|---|---|---|---|---|---|---|---|---|---|---|---|
| *2s_vs_1sc* (10k) | 0 | **93.4(3.1)** | 90.4(5.4) | 86.8(4.4) | 63.8(15.2) | *3s_vs_5z* (20k) | 0 | **93.5(1.8)** | 85.1(3.4) | 55.0(12.5) | 51.0(8.8) |
| | 0.2 | **93.1(2.9)** | 89.9(4.5) | 82.8(5.1) | 61.3(14.8) | | 0.2 | **91.0(1.7)** | 78.0(2.8) | 36.0(9.0) | 31.0(7.0) |
| | 0.4 | **93.5(3.3)** | 90.1(4.0) | 81.4(4.2) | 58.9(13.6) | | 0.4 | **85.0(2.2)** | 68.0(3.3) | 23.0(6.0) | 17.0(5.0) |
| | 0.6 | **93.1(2.5)** | 88.4(5.5) | 80.2(4.1) | 52.2(10.6) | | 0.6 | **74.0(1.5)** | 48.0(2.8) | 5.0(0.0) | 2.0(0.0) |
| | 0.8 | **93.2(3.6)** | 89.2(4.1) | 78.1(3.5) | 48.2(8.7) | | 0.8 | **46.0(3.5)** | 14.0(3.4) | 0.0(0.0) | 0.0(0.0) |
| | 1.0 | **93.0(3.8)** | 88.4(4.2) | 72.2(3.2) | 42.2(7.0) | | 1.0 | 0.0(0.0) | 0.0(0.0) | 0.0(0.0) | 0.0(0.0) |
| *3s_vs_3z* (20k) | 0 | **95.2(2.7)** | 88.2(3.6) | 83.2(5.6) | 78.0(10.2) | *8m* (15k) | 0 | **90.8(2.2)** | 83.5(3.8) | 63.0(7.6) | 36.0(7.9) |
| | 0.2 | **94.3(3.0)** | 88.0(2.6) | 82.0(5.0) | 74.0(9.7) | | 0.2 | **93.5(3.6)** | 81.0(2.9) | 60.0(6.5) | 31.0(7.2) |
| | 0.4 | **94.0(3.3)** | 84.0(2.9) | 70.5(4.3) | 61.0(8.5) | | 0.4 | **91.0(4.3)** | 73.0(3.4) | 51.0(4.8) | 24.0(4.3) |
| | 0.6 | **93.0(2.0)** | 72.0(2.4) | 60.2(3.4) | 51.0(6.7) | | 0.6 | **92.0(3.6)** | 66.0(2.9) | 33.0(3.2) | 5.0(1.4) |
| | 0.8 | **84.0(2.7)** | 56.0(2.0) | 24.2(3.2) | 17.0(4.2) | | 0.8 | **58.0(3.4)** | 39.0(2.5) | 22.0(2.1) | 0.0(0.0) |
| | 1.0 | **35.0(2.0)** | 11.0(3.0) | 1.0(1.0) | 0.0(0.0) | | 1.0 | **33.0(2.5)** | 13.0(3.5) | 3.0(0.2) | 0.0(0.0) |
| *2s3z* (15k) | 0 | **98.1(0.8)** | 87.1(3.2) | 67.1(11.3) | 70.9(11.9) | *1c3s5z* (20k) | 0 | **97.6(2.8)** | 84.9(3.0) | 63.0(12.9) | 36.0(9.2) |
| | 0.2 | **95.0(1.0)** | 84.0(3.4) | 64.0(12.2) | 67.0(10.9) | | 0.2 | **98.0(2.5)** | 85.0(2.0) | 59.0(11.6) | 34.0(8.3) |
| | 0.4 | **97.0(2.4)** | 83.0(3.1) | 56.0(10.9) | 63.0(8.8) | | 0.4 | **97.0(2.4)** | 82.0(3.5) | 56.5(8.7) | 31.2(6.6) |
| | 0.6 | **95.0(1.1)** | 76.0(2.6) | 47.0(9.6) | 53.0(6.4) | | 0.6 | **95.0(1.7)** | 74.0(3.0) | 42.0(4.2) | 6.0(4.6) |
| | 0.8 | **92.7(1.9)** | 62.0(2.2) | 39.0(8.8) | 40.0(5.6) | | 0.8 | **91.0(1.3)** | 46.0(2.6) | 16.0(1.0) | 0.0(0.0) |
| | 1.0 | **79.3(2.2)** | 43.0(3.1) | 15.0(3.4) | 16.0(3.2) | | 1.0 | **6.0(2.6)** | 1.0(3.6) | 0.0(0.0) | 0.0(0.0) |
| *3s_vs_4z* (20k) | 0 | **96.9(0.8)** | 96.1(1.3) | 81.0(11.4) | 64.0(32.7) | *corridor* (50k) | 0 | **85.0(3.6)** | 60.2(19.0) | 27.0(7.3) | 39.0(9.2) |
| | 0.2 | **93.0(0.8)** | 91.0(2.7) | 76.0(10.3) | 58.0(28.9) | | 0.2 | **79.0(3.4)** | 51.0(12.1) | 24.0(6.7) | 31.0(7.4) |
| | 0.4 | **94.0(2.1)** | 82.0(4.2) | 66.0(9.4) | 42.0(20.1) | | 0.4 | **76.0(3.5)** | 41.0(8.6) | 18.0(5.8) | 21.5(4.7) |
| | 0.6 | **86.0(1.4)** | 67.0(4.7) | 44.0(6.7) | 26.0(10.6) | | 0.6 | **71.0(3.8)** | 32.0(5.1) | 11.0(3.2) | 10.0(2.3) |
| | 0.8 | **58.0(1.6)** | 37.0(3.3) | 11.0(3.5) | 4.0(4.2) | | 0.8 | **38.0(2.2)** | 11.0(4.7) | 4.0(0.4) | 3.0(0.2) |
| | 1.0 | 0.0(0.0) | 0.0(0.0) | 0.0(0.0) | 0.0(0.0) | | 1.0 | 0.0(0.0) | 0.0(0.0) | 0.0(0.0) | 0.0(0.0) |

**Table A3** The experimental result of all methods in 1000 random matches under different observation loss probabilities $p_{\text{loss}}$ on different maps in MAMuJoCo. Each map name is followed by the number of fine-tuning steps in parentheses. Results are reported as the average step return (standard deviation) aggregated over three random seeds, with the highest-performing entry highlighted in bold.

| **Map** | $p_{\text{loss}}$ | RMIOv2 | RMIO [24] | MAG [19] | MAMBA [18] |
|---|---|---|---|---|---|
| *HalfCheetah* (50k) | 0 | **5.08(0.46)** | 4.63(0.43) | 4.07(0.37) | 3.71(0.69) |
| | 0.2 | **4.97(0.44)** | 4.56(0.50) | 3.27(0.13) | 2.97(0.48) |
| | 0.4 | **5.04(0.32)** | 4.01(0.40) | 2.39(0.08) | 2.16(0.32) |
| | 0.6 | **4.92(0.45)** | 3.24(0.52) | 2.12(0.05) | 1.88(0.22) |
| | 0.8 | **3.81(0.26)** | 2.82(0.31) | 1.43(0.03) | 1.07(0.14) |
| | 1.0 | **0.38(0.18)** | 0.14(0.20) | -0.21(0.01) | -0.23(0.04) |
| 4**Swimmer* (50k) | 0 | 0.32(0.01) | **0.34(0.01)** | 0.29(0.01) | 0.26(0.01) |
| | 0.2 | 0.31(0.01) | **0.32(0.01)** | 0.24(0.01) | 0.21(0.01) |
| | 0.4 | **0.31(0.01)** | 0.30(0.01) | 0.21(0.01) | 0.17(0.01) |
| | 0.6 | **0.29(0.01)** | 0.26(0.01) | 0.16(0.01) | 0.11(0.01) |
| | 0.8 | **0.19(0.01)** | 0.12(0.01) | 0.04(0.01) | 0.02(0.01) |
| | 1.0 | **0.01(0.01)** | -0.02(0.01) | -0.03(0.01) | -0.03(0.01) |
| 10**Swimmer* (50k) | 0 | **0.35(0.02)** | 0.34(0.02) | 0.14(0.14) | 0.22(0.07) |
| | 0.2 | **0.34(0.02)** | 0.31(0.01) | 0.13(0.11) | 0.20(0.04) |
| | 0.4 | **0.33(0.01)** | 0.28(0.01) | 0.08(0.08) | 0.18(0.02) |
| | 0.6 | **0.30(0.01)** | 0.23(0.01) | 0.02(0.07) | 0.12(0.01) |
| | 0.8 | **0.23(0.01)** | 0.13(0.01) | -0.03(0.03) | 0.04(0.01) |
| | 1.0 | **0.09(0.01)** | 0.06(0.01) | -0.06(0.01) | -0.04(0.01) |
| 4**Ant* (50k) | 0 | **2.44(0.42)** | 2.13(0.35) | 1.88(0.55) | 1.31(0.23) |
| | 0.2 | **2.34(0.31)** | 2.03(0.30) | 1.63(0.44) | 1.18(0.23) |
| | 0.4 | **2.29(0.19)** | 1.87(0.11) | 1.47(0.34) | 1.00(0.18) |
| | 0.6 | **1.98(0.14)** | 1.34(0.10) | 1.02(0.33) | 0.66(0.12) |
| | 0.8 | **1.41(0.09)** | 0.78(0.06) | 0.44(0.24) | 0.21(0.12) |
| | 1.0 | **0.76(0.06)** | 0.45(0.02) | 0.02(0.01) | 0.01(0.01) |

## Appendix B Hyperparameter configurations

As shown in Table B1 and B2, this section presents the hyperparameter settings of the RMIOv2 algorithm, including the structural parameters of the world model and policy model, as well as the parameter settings during the training process.

**Table B1** Model configuration parameters.

| | **Parameter** | **Value** |
|---|---|---|
| **World Model** | ENCODER_HIDDEN | 256 |
| | ENCODER_LAYERS | 2 |
| | OBS_EMBED | 256 |
| | ATTENTION_HIDDEN | 256 |
| | ATTENTION_LAYERS | 3 |
| | POSTERIOR_HIDDEN | 256 |
| | POSTERIOR_LAYERS | 2 |
| | PRIOR_HIDDEN | 256 |
| | PRIOR_LAYERS | 2 |
| | N_CATEGORICALS | 32 |
| | N_CLASSES | 32 |
| | STOCHASTIC ($K$) | $32 \times 32 = 1024$ |
| | DETERMINISTIC | 256 |
| | FEAT | $1024 + 256 = 1280$ |
| | OBS_HIDDEN | 256 |
| | OBS_LAYERS | 3 |
| | REWARD_HIDDEN | 256 |
| | REWARD_LAYERS | 2 |
| | PCONT_HIDDEN | 256 |
| | PCONT_LAYERS | 2 |
| | AV_ACTION_HIDDEN | 256 |
| | AV_ACTION_LAYERS | 2 |
| **Policy** | VALUE_HIDDEN | 64 |
| | VALUE_LAYERS | 2 |
| | VALUE_ATTENTION | 256 |
| | ACTION_HIDDEN | 256 |
| | ACTION_LAYERS | 2 |

**Table B2** Training configuration parameters.

| | **Parameter** | **Value** |
|---|---|---|
| **World Model Training** | MODEL_BUFFER_CAPACITY | 240,000 |
| | MODEL_MIN_BUFFER_SIZE | 500 |
| | MODEL_EPOCHS | 60 |
| | MODEL_BATCH_SIZE | 60 |
| | SEQ_LENGTH | 20 |
| | H_BASE | 5 |
| | H_MIN | 3 |
| | H_MAX | 7 |
| | ALPHA | 80 |
| | $p_{\text{mask}}$ | 0.5 |
| | HORIZON | 10 |
| | MODEL_LR | $5 \times 10^{-4}$ |
| | GRAD_CLIP | 5 |
| **Policy Optimization** | POLICY_BUFFER_CAPACITY | 201600 |
| | POLICY_MIN_BUFFER_SIZE | 500 |
| | ROLLOUT_LENGTH | 15 |
| | POLICY_EPOCHS | 20 |
| | POLICY_BATCH_SIZE | 100 |
| | ENTROPY | 0.001 |
| | ENTROPY_ANNEALING | 0.99998 |
| | GAMMA | 0.99 |
| | DISCOUNT | 0.99 |
| | DISCOUNT_LAMBDA | 0.95 |
| | TAU | 0.1 |
| | GRAD_CLIP_POLICY | 5 |
| | ACTOR_LR | $5 \times 10^{-4}$ |
| | CRITIC_LR | $5 \times 10^{-4}$ |

## Appendix C Sensitivity analysis for key parameters

To verify how core parameters affect RMIOv2's performance, this section conducts sensitivity analysis on 5 parameters tied to the model's core mechanisms (masked fine-tuning, dynamic reward modeling, RSSM structure, cross-agent Transformer fusion), with results in Table C1. $p_{\text{mask}}$ is evaluated by AUC across $p_{\text{loss}} \in [0, 1]$ (measuring observation loss robustness, as in Table A2), while other parameters use the average win rate of 3 independent trials for statistical reliability.

**Table C1** Parameter Sensitivity Analysis for RMIOv2 on $2s3z$ Task. $p_{\text{mask}}$ uses AUC (Area Under Curve, across $p_{\text{loss}} \in [0, 1]$) to quantify robustness to observation loss; other parameters use average win rate from 3 independent experiments.

| Parameter | Default Value | Adjusted Value | Performance Metric (vs. Default) |
|---|---|---|---|
| $p_{\text{mask}}$ | 0.5 | 0.125 | AUC ↓ 41.5% (93.8%→54.9%, across $p_{\text{loss}} \in [0, 1]$) |
| | | 0.25 | AUC ↓ 12.9% (93.8%→81.7%, across $p_{\text{loss}} \in [0, 1]$) |
| | | 0.375 | AUC ↓ 3.41% (93.8%→90.6%, across $p_{\text{loss}} \in [0, 1]$) |
| | | 0.75 | AUC ↓ 1.49% (93.8%→92.4%, across $p_{\text{loss}} \in [0, 1]$) |
| $H_{\text{base}}$ | 5 | 3 | Avg. Win Rate ↓ 1.83% (98.1%→96.3%) |
| | | 7 | Avg. Win Rate ↓ 4.69% (98.1%→93.5%) |
| $K$ | 1024 (32×32) | 576 (24×24) | Avg. Win Rate ↓ 5.61% (98.1%→92.6%) |
| | | 2304 (48×48) | Avg. Win Rate ↓ 2.34% (98.1%→95.8%) |
| ATTENTION_HIDDEN | 256 | 128 | Avg. Win Rate ↓ 8.56% (98.1%→89.7%) |
| | | 512 | Avg. Win Rate ↓ 3.57% (98.1%→94.6%) |
| ATTENTION_LAYERS | 3 | 2 | Avg. Win Rate ↓ 6.42% (98.1%→91.8%) |
| | | 4 | Avg. Win Rate ↓ 3.67% (98.1%→94.5%) |

As the key parameter for masked fine-tuning, $p_{\text{mask}}$ (default 0.5) shows notable AUC changes when adjusted: $p_{\text{mask}} = 0.125$ (insufficient masking) leads to a 41.5% AUC drop (93.8%→54.9%) due to under-trained cross-agent observation reconstruction, while $p_{\text{mask}} = 0.75$ (excessive masking) only causes a 1.49% drop—thanks to the simple agent interactions of $2s3z$ task, which mitigate overfitting. Moderate masking (0.375-0.5) balances robustness training and overfitting avoidance, aligning with the paper's optimal range. For $H_{\text{base}}$ (default 5), adjusting $H_{\text{base}}$ to 3 reduces the average win rate by 1.83% (98.1%→96.3%) for failing to capture multi-step reward trends, while 7 causes a 4.69% drop (98.1%→93.5%) due to delayed reward feedback—confirming that $H_{\text{base}} = 5$ balances local detail and long-term trend capture. The RSSM stochastic latent dimensionality $K$ (default 1024=32×32) affects state representation: $K = 576 = 24 \times 24$ leads to a 5.61% win rate drop (98.1%→92.6%) for incomplete encoding of agent interactions, while $K = 2304(48 \times 48)$ reduces win rate by 2.34% (98.1%→95.8%)—proving $K = 1024$ balances representation capability and computational efficiency, which is consistent with Appendix D's cost analysis. For cross-agent Transformer parameters, ATTENTION_HIDDEN (default 256) dropping to 128 causes an 8.56% win rate drop (98.1%→89.7%) for insufficient feature extraction, while rising to 512 only reduces by 3.57% (98.1%→94.6%) with limited marginal gain; ATTENTION_LAYERS (default 3) falling to 2 leads to a 6.42% drop (98.1%→91.8%) for incomplete multi-scale fusion, while rising to 4 causes a 3.67% drop (98.1%→94.5%) for unnecessary complexity.

## Appendix D Computational and parametric analysis of dual Transformer modules

To quantify the computational and parametric overhead of the dual cross-agent Transformer modules (observation-fusion and action-fusion), we have conducted experiments across four SMAC scenarios ($2s\_vs\_1sc$, $3s\_vs\_3z$, $2s3z$, $1c3s5z$), comparing the FLOPs (floating-point operations), parameter counts, and GPU Memory (GM) of RMIOv2 with its ablation variants (removing one or both modules). Table D1 summarizes the tool-measured results, where FLOPs are calculated per forward pass, parameters represent the total count for the entire world model $\mathcal{M}$, and GM is measured on a single NVIDIA RTX 4090 GPU under FP32 precision (inference phase).

**Table D1** Computational, parametric and memory comparison of RMIOv2 variants across SMAC scenarios. FLOPs are calculated per forward pass; parameters are counted for the entire world model $\mathcal{M}$; GM (GPU Memory) is measured on a single NVIDIA RTX 4090 GPU under FP32 precision (inference phase).

| Model Variant | $2s\_vs\_1sc$ | | | $3s\_vs\_3z$ | | | $2s3z$ | | | $1c3s5z$ | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FLOPs (M) | Params (M) | GM (GB) | FLOPs (M) | Params (M) | GM (GB) | FLOPs (M) | Params (M) | GM (GB) | FLOPs (M) | Params (M) | GM (GB) |
| Full Model | 13.23 | 3.84 | 0.68 | 19.87 | 3.85 | 0.72 | 33.23 | 3.87 | 0.74 | 60.23 | 3.91 | 0.77 |
| w/o Obs-Fusion | 12.30 | 3.37 | 0.66 | 18.48 | 3.38 | 0.71 | 30.90 | 3.40 | 0.72 | 56.05 | 3.45 | 0.75 |
| w/o Act-Fusion | 12.13 | 3.56 | 0.65 | 18.23 | 3.58 | 0.70 | 30.48 | 3.60 | 0.71 | 55.29 | 3.64 | 0.73 |
| w/o Both | 11.20 | 3.10 | 0.64 | 16.83 | 3.11 | 0.69 | 28.16 | 3.13 | 0.70 | 51.11 | 3.18 | 0.71 |

Focusing on the $1c3s5z$ scenario, the results reveal the specific resource proportion of each Transformer module: The observation-fusion module accounts for 6.95% of total FLOPs and 11.86% of total parameters (reflecting heavier parametric demand for high-dimensional observation processing), while the action-fusion module accounts for 8.20% of FLOPs but only 6.89% of parameters (lower parametric demand for low-dimensional action embeddings). Together, the dual modules account for only 15.15% of total FLOPs and 18.74% of total parameters—a modest overhead that does not compromise scalability, and a pattern that holds consistently across other scenarios. Across all four scenarios, GM values follow a consistent, mild upward trend with increasing scenario complexity (e.g., 0.64-0.68GB for $2s\_vs\_1sc$, while 0.71-0.77GB for $1c3s5z$), directly mirroring changes in model complexity. Since this complexity increase is primarily driven by a larger number of agents, the gradual resource growth curve means the additional computational and storage burden from more agents remains well within acceptable limits during inference.

## Appendix E Experiment analysis

### Appendix E.1 Attention weight visualization of cross-agent Transformer modules

To intuitively illustrate how the observation-fusion module prioritizes inter-agent information during adaptation to observation loss, we visualize the attention weights of all 8 parallel attention heads in the cross-agent Transformer (consistent with the model configuration in Table B1) before and after masked fine-tuning. The experimental scenario is configured as the $1c3s5z$ map and we randomly select timestep $t = 20$ as the analysis sample—featuring 9 agents total, with only Agent 1, 3, and 5 not experiencing observation loss—to explicitly examine the attention patterns of each agent toward one another.

Figure E1 presents the attention weight distribution across all 8 attention heads of the observation-fusion Transformer before masked fine-tuning. Each subplot corresponds to one attention head, with the x-axis and y-axis denoting the indices of agents in the Key and Query roles, respectively, and color intensity representing the magnitude of attention weight. Prior to fine-tuning, the attention weights are relatively scattered across all agents, with no obvious focus on Agent 1, 3, and 5 (the ones with complete observations). This scattered distribution reflects the model's inability to prioritize information from observation-reliable agents, limiting its capacity to leverage cross-agent cues for missing observation reconstruction.

Figure E2 shows the attention weight distribution of the same 8 attention heads after masked fine-tuning (with $p_{\text{mask}} = 0.5$ as in Section 5.2). A clear shift in attention patterns is observed: All attention heads consistently allocate higher weights to Agent 1, 3, and 5—these are the agents with no observation loss in the test scenario. This targeted attention prioritization is a direct result of masked fine-tuning: The model learns to focus on agents with reliable observational information, as these agents provide critical cues for cross-agent information fusion. By amplifying attention to Agent 1, 3, and 5, the observation-fusion module efficiently aggregates valid information to reconstruct observations for agents experiencing loss, which aligns with the core design goal of enhancing robust decision-making under incomplete observations.

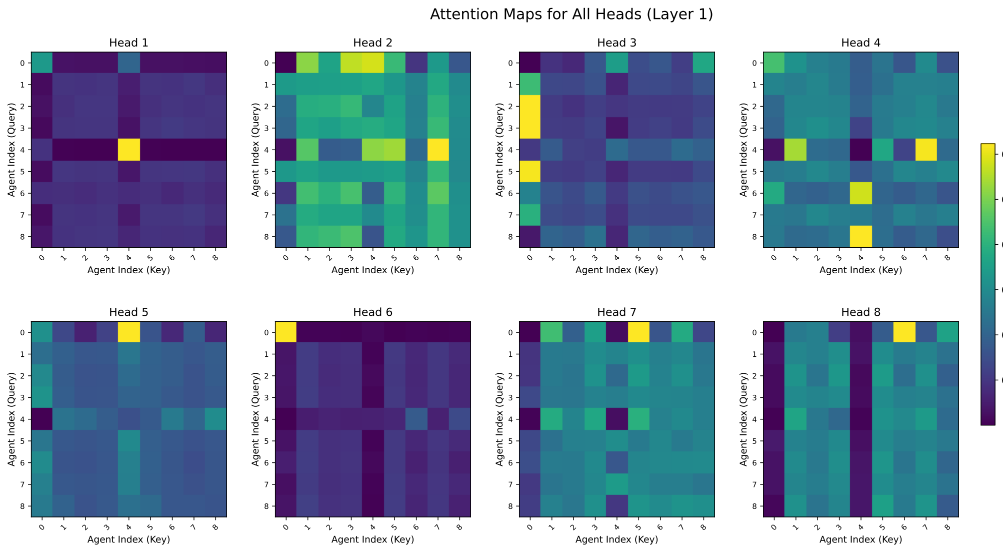

**Figure E1** Attention weight heatmaps of the cross-agent Transformer (observation-fusion module) before masked fine-tuning. Each subplot corresponds to one of the 8 parallel attention heads. The x-axis/y-axis represents agents in the Key/Query role, and color intensity indicates attention weight. No obvious focus on specific agents is observed.

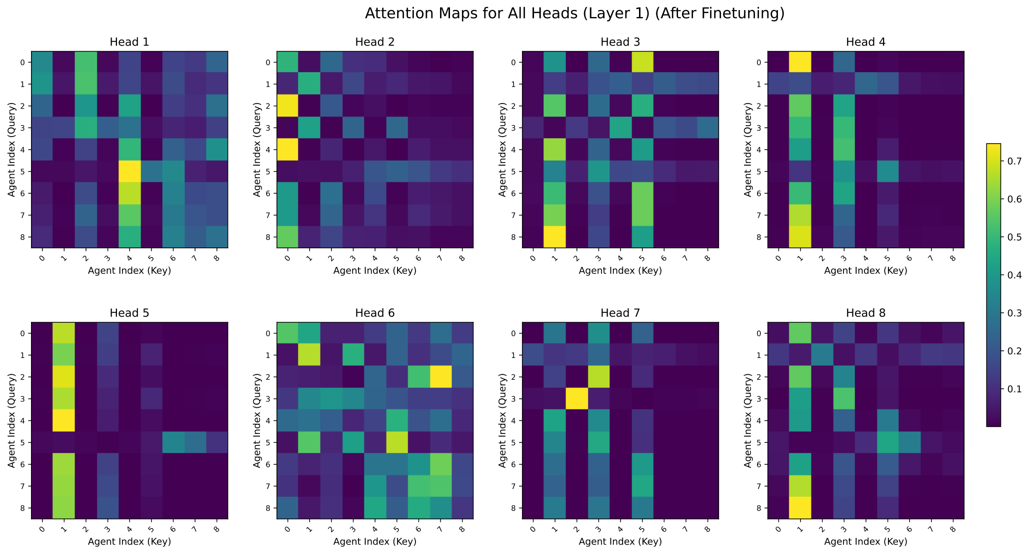

**Figure E2** Attention weight heatmaps of the cross-agent Transformer (observation-fusion module) after masked fine-tuning. All 8 attention heads consistently allocate higher weights to agents 1, 3, and 5 (agents with no observation loss), reflecting the model's learned ability to prioritize information from observation-reliable agents.

## Appendix E.2 Visualization of dynamic reward temporal smoothing

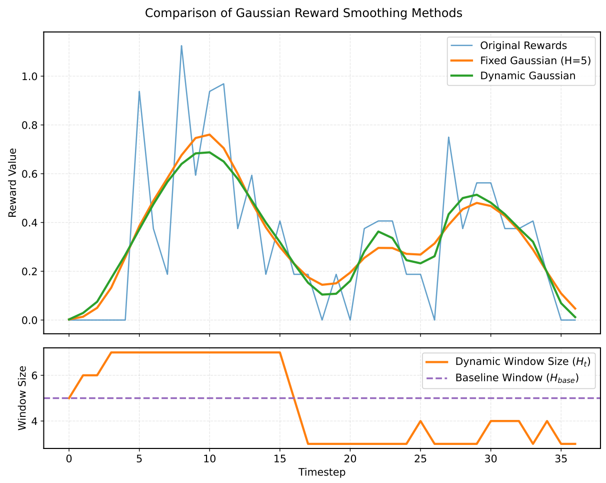

**Figure E3** Comparison of reward smoothing methods in high-variance episodes. The top panel shows original rewards alongside fixed-window (H=5) and dynamic-window smoothing results. The bottom panel illustrates the adaptive window size $H_t$ responding to local reward variance, with $H_{\text{base}} = 5$ providing the baseline for variance calculation. The dynamic approach automatically widens the smoothing window in high-variance regions to capture long-term trends while maintaining finer granularity in stable periods.

To validate the effectiveness of RMIOv2's dynamic reward temporal smoothing mechanism (Section 4.2) in mitigating reward modeling errors under multi-agent interaction complexity, we visualize the smoothing results of team rewards in high-variance episodes (e.g., the SMAC $1c3s5z$ scenario, where agent combat interactions cause abrupt reward fluctuations) and compare them with fixed-window Gaussian smoothing.

Figure E3 presents a side-by-side analysis of three reward sequences: the original unsmoothed rewards, rewards processed with fixed-window Gaussian smoothing (window size $H = 5$), and rewards processed with RMIOv2's dynamic-window smoothing. The top panel of Figure E3 shows that the original reward sequence exhibits significant volatility—sharp spikes (e.g., at Timesteps 7, 12) and sudden drops (e.g., at Timesteps 8, 10) occur frequently, which can mislead the world model into learning noisy temporal dynamics and destabilize policy updates. Fixed-window smoothing ($H = 5$) partially reduces this noise but fails to adapt to context: it over-smooths in stable regions (e.g., Timesteps 15-25), blurring fine-grained reward trends that reflect subtle agent coordination; in high-variance regions (e.g., Timesteps 5-15), it cannot capture long-term reward dependencies, leading to incomplete modeling of combat outcome correlations.

In contrast, RMIOv2's dynamic smoothing adjusts the window size $H_t$ in real time based on local reward variance (calculated using the baseline window $H_{\text{base}} = 5$, as defined in Eq. (12)), as shown in the bottom panel of Figure E3. When reward variance is low (e.g., Timesteps 20-35), $H_t$ shrinks to 3-4, preserving fine-grained temporal details that encode short-term agent collaboration (e.g., minor damage-dealing events). When variance surges (e.g., at Timesteps 12-20, corresponding to large-scale team combat), $H_t$ automatically expands to 6–7, thus integrating longer-range reward information to capture the cumulative impact of combat sequences (e.g., consecutive enemy eliminations). This adaptability ensures that the smoothed reward sequence retains both local details and global trend consistency, as evidenced by the stable yet responsive curve in the top panel. This effectiveness is further supported by the ablation results in Section 6.3.1.